



UNIVERSIDADE  
DE VIGO

ESCOLA SUPERIOR DE ENXEÑARÍA

Memoria do Traballo de Fin de Grao que presenta

**D. Julián Jiménez González**

para a obtención do Título de Graduado en Enxeñaría Informática

**SuperCars: Desarrollo de videojuego para dispositivos móviles**



Xaneiro, 2015

**Traballo de Fin de Grao N°:** EI 14/15 - 010

**Titor/a:** Alma María Gómez Rodríguez

**Área de coñecemento:** Linguaxes e Sistemas Informáticos

**Departamento:** Informática



# Índice

<b>1.</b>	<b>INTRODUCCIÓN.....</b>	<b>4</b>
1.1.	DESCRIPCIÓN DEL SISTEMA .....	4
1.2.	ARQUITECTURA DE LA APLICACIÓN .....	4
<b>2.</b>	<b>ANÁLISIS .....</b>	<b>4</b>
2.1.	ESPECIFICACIÓN DE REQUISITOS FUNCIONALES.....	4
2.2.	MODELO DE DOMINIO .....	4
2.3.	DESCRIPCIÓN DE CASOS DE USO .....	5
2.3.1.	<i>Diagrama de Casos de Uso .....</i>	<i>5</i>
2.3.2.	<i>Descripción detallada de Casos de Uso.....</i>	<i>6</i>
2.3.2.1.	Iniciar partida .....	6
2.3.2.2.	Iniciar carrera .....	6
2.3.2.3.	Mover vehículo .....	6
2.3.2.4.	Terminar carrera .....	7
2.3.2.5.	Finalizar partida.....	7
2.3.2.6.	Pausar carrera .....	7
2.3.2.7.	Reanudar carrera .....	8
2.3.2.8.	Salir de carrera .....	8
2.3.2.9.	Salir del juego.....	8
<b>3.</b>	<b>DISEÑO .....</b>	<b>8</b>
3.1.	DIAGRAMAS DE CLASES PARCIALES .....	9
3.1.1.	<i>AppDelegate .....</i>	<i>9</i>
3.1.2.	<i>MainMenu .....</i>	<i>10</i>
3.1.3.	<i>RaceConf.....</i>	<i>10</i>
3.1.4.	<i>RankingMenu.....</i>	<i>11</i>
3.1.5.	<i>Race .....</i>	<i>11</i>
3.1.6.	<i>RaceMenu.....</i>	<i>12</i>
3.1.7.	<i>EndRace .....</i>	<i>12</i>
3.1.8.	<i>Diagrama de clases total.....</i>	<i>12</i>
3.2.	DIAGRAMAS DE SECUENCIA DEL SISTEMA .....	14
3.2.1.	<i>Iniciar partida .....</i>	<i>14</i>
3.2.2.	<i>Iniciar carrera .....</i>	<i>14</i>
3.2.3.	<i>Mover vehículo .....</i>	<i>15</i>
3.2.4.	<i>Terminar Carrera .....</i>	<i>15</i>
3.2.5.	<i>Finalizar partida.....</i>	<i>15</i>
3.2.6.	<i>Pausar carrera .....</i>	<i>16</i>
3.2.7.	<i>Reanudar carrera.....</i>	<i>16</i>
3.2.8.	<i>Salir de carrera .....</i>	<i>16</i>
3.2.9.	<i>Salir del juego .....</i>	<i>16</i>

## 1. Introducción

---

### 1.1. Descripción del sistema

SuperCars es un videojuego de carreras en 2D. En él, el jugador debe mover el vehículo a izquierda y derecha para evitar los obstáculos que van apareciendo en la carretera, así como adelantar a los oponentes. Se pretende que la aleatoriedad en la generación de obstáculos y oponentes provoque situaciones distintas en cada partida, así como el incremento de dificultad.

### 1.2. Arquitectura de la aplicación

## 2. Análisis

---

### 2.1. Especificación de requisitos funcionales

En este apartado se reflejan todos los requisitos funcionales

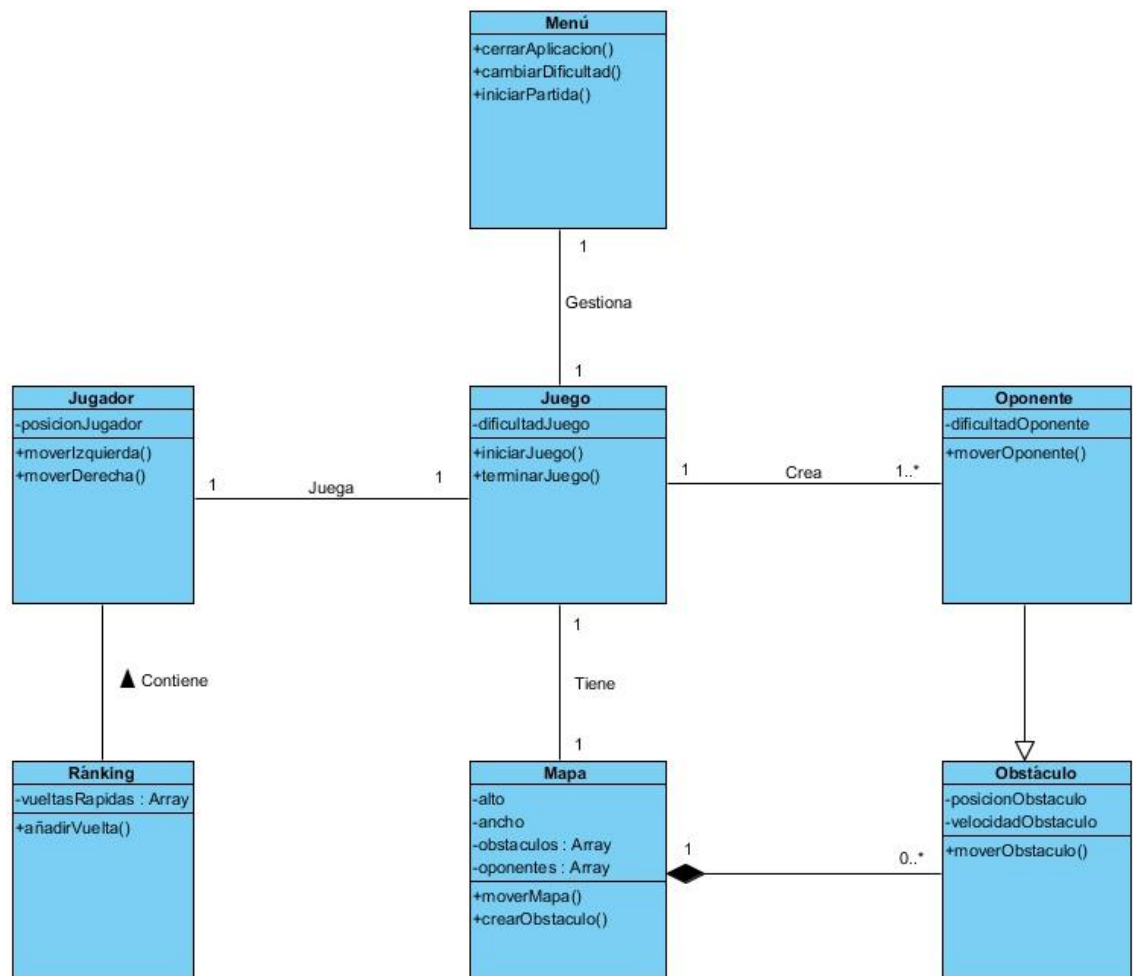
### 2.2. Modelo de Dominio

En el Modelo de Dominio se reflejan las clases (conceptuales) que se consideran significativas para el dominio del problema.

Estas clases no tienen por qué coincidir con las implementadas finalmente, ya que éstas últimas pueden variar en función del entorno de desarrollo e incluso del de despliegue.

La lógica y funcionalidades descritas en estas clases sí deben mantenerse.

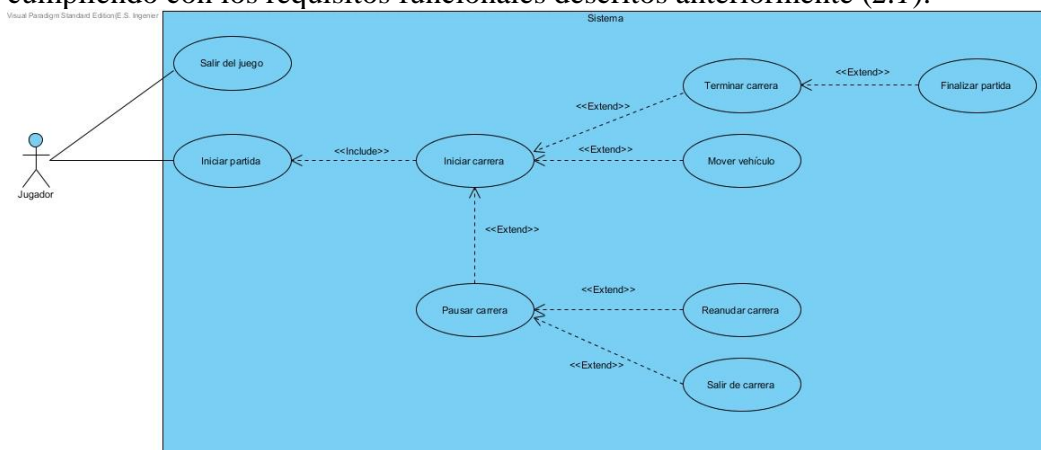
La versión final del Modelo de Dominio que refleja el diseño real de la aplicación se puede encontrar en el apartado 3.1.8.



## 2.3. Descripción de Casos de Uso

### 2.3.1. Diagrama de Casos de Uso

Este diagrama representa las interacciones del usuario con el sistema, cumpliendo con los requisitos funcionales descritos anteriormente (2.1).



A continuación se realiza una descripción detallada de los Casos de Uso, que incluye:

- Breve descripción.

- Precondiciones.
- Postcondiciones.
- Flujo de eventos.
- Flujo alternativo (opcional).

Se obvia el actor debido a que en todos ellos es el jugador.

### 2.3.2. Descripción detallada de Casos de Uso

#### 2.3.2.1. Iniciar partida

Iniciar partida			
Descripción breve	El jugador pulsa el botón <i>Play</i> en <i>MainMenu</i> y la escena cambia a <i>RaceConf</i>		
Precondiciones	La escena activa es <i>MainMenu</i>		
Postcondiciones	La escena activa es <i>RaceConf</i>		
Flujo Principal	Jugador		Sistema
	1	Pulsa el botón <i>Play</i>	
	2		Cambia escena activa de <i>MainMenu</i> a <i>RaceConf</i>

#### 2.3.2.2. Iniciar carrera

Iniciar carrera			
Descripción breve	El jugador configura las opciones de carrera y ésta da comienzo de acuerdo con ellas		
Precondiciones	La escena activa es <i>RaceConf</i>		
Postcondiciones	La escena activa es <i>Race</i>		
Flujo Principal	Jugador		Sistema
	1		Muestra parámetros de configuración de la carrera
	2	Ajusta los parámetros de configuración	
	3		Recoge la configuración introducida e inicia la carrera de acuerdo a dicha configuración

#### 2.3.2.3. Mover vehículo

Mover vehículo		
Descripción breve	El jugador pulsa un botón de dirección y el vehículo se mueve al sitio correspondiente	
Precondiciones	La escena activa es <i>Race</i>	
Postcondiciones	El vehículo se ha movido a la posición deseada	
Flujo Principal	Jugador	
	Sistema	
	1	Pulsa botón de movimiento <i>leftArrow</i> o <i>rightArrow</i>
	2	Identifica cuál de los

			botones ha sido pulsado
	3		Mueve el vehículo al lado correspondiente
	4		Se queda a la espera de una nueva interacción
Flujo Alternativo [A3]	<b>Jugador</b>		<b>Sistema</b>
	1		Si está a la izquierda y se ha pulsado <i>leftArrow</i> , no hace nada. Vuelve al paso 4
Flujo alternativo [A3]	<b>Jugador</b>		<b>Sistema</b>
	1		Si está a la derecha y se ha pulsado <i>rightArrow</i> , no hace nada. Vuelve al paso 4

#### 2.3.2.4. Terminar carrera

Terminar carrera			
Descripción breve	El jugador alcanza la línea de meta de la última vuelta y la carrera finaliza		
Precondiciones	La escena activa es <i>Race</i>		
Postcondiciones	La escena activa es <i>EndRace</i>		
Flujo Principal	<b>Jugador</b>		<b>Sistema</b>
	1	Alcanza la línea de meta	
	2		Finaliza la carrera
	3		Cambia escena activa de <i>Race</i> a <i>EndRace</i>

#### 2.3.2.5. Finalizar partida

Finalizar partida			
Descripción breve	El jugador vuelve a <i>MainMenu</i> pulsando <i>back</i> en la escena <i>EndRace</i> , una vez terminada la carrera		
Precondiciones	La escena activa es <i>EndRace</i>		
Postcondiciones	La escena activa es <i>MainMenu</i>		
Flujo Principal	<b>Jugador</b>		<b>Sistema</b>
	1	Pulsa <i>back</i>	
	2		Cambia escena activa de <i>EndRace</i> a <i>MainMenu</i>

#### 2.3.2.6. Pausar carrera

Pausar carrera			
Descripción breve	El jugador pulsa el botón <i>menú</i> en la escena <i>Race</i> y se pausa el juego para mostrar <i>RaceMenu</i>		
Precondiciones	La escena activa es <i>Race</i>		
Postcondiciones	La escena activa es <i>RaceMenu</i>		
Flujo Principal	<b>Jugador</b>		<b>Sistema</b>
	1	Pulsa botón <i>menu</i>	

	2		Pausa escena <i>Race</i>
	3		Muestra escena <i>RaceMenu</i>

#### 2.3.2.7. Reanudar carrera

Reanudar carrera			
Descripción breve	El jugador pulsa <i>resume</i> en <i>RaceMenu</i> y se reanuda la carrera		
Precondiciones	La escena activa es <i>RaceMenu</i>		
Postcondiciones	La escena activa es <i>Race</i>		
Flujo Principal	<b>Jugador</b>		<b>Sistema</b>
	1	Pulsa <i>resume</i>	
	3		Cambia escena activa de <i>RaceMenu</i> a <i>Race</i>

#### 2.3.2.8. Salir de carrera

Salir de carrera			
Descripción breve	El jugador pulsa el botón <i>Exit</i> en <i>RaceMenu</i> y la escena cambia a <i>MainMenu</i>		
Precondiciones	La escena activa es <i>RaceMenu</i>		
Postcondiciones	La escena activa es <i>MainMenu</i>		
Flujo Principal	<b>Jugador</b>		<b>Sistema</b>
	1	Pulsa el botón <i>exit</i>	
	2		Finaliza la escena <i>Race</i>
	3		Cambia escena activa de <i>RaceMenu</i> a <i>MainMenu</i>

#### 2.3.2.9. Salir del juego

Salir del juego			
Descripción breve	El jugador pulsa el botón <i>exit</i> de <i>MainMenu</i> y la aplicación se cierra		
Precondiciones	La escena activa es <i>MainMenu</i>		
Postcondiciones	Aplicación cerrada correctamente		
Flujo Principal	<b>Jugador</b>		<b>Sistema</b>
	1	Pulsa <i>exit</i> en <i>MainMenu</i>	
	2		Cierra la aplicación

## 3. Diseño

Las variaciones en el diseño con respecto al análisis (2) están producidas por la adecuación del desarrollo al *framework* Cocos2d-x.

A continuación se muestran las clases parciales finales, así como el diagrama de clases total.



### 3.1. Diagramas de clases parciales

En este apartado se detallan las clases parciales del sistema, que se pueden identificar con una escena del juego (exceptuando AppDelegate).

Todas las clases tienen su diagrama parcial, una breve explicación y sus métodos y variables más importantes.

Existen ciertas variables y métodos que son comunes a casi todas las clases, por lo que se detallan aquí:

- *origin*: variable de tipo *Vec2* que contiene las coordenadas X e Y del origen.
- *visibleSize*: variable de tipo *Size* que contiene la altura y el ancho de la pantalla.
- *createScene()*: es el constructor de la escena, por lo que devuelve un objeto escena básico.
- *init()*: función que se ejecuta siempre y contiene las llamadas principales de la clase. Se puede considerar como una función *main*<sup>1</sup>, por lo que el resto de métodos son llamados (directa o indirectamente) desde éste.
- *createMenu()*: crea y coloca los botones del menú correspondiente a esa escena.

#### 3.1.1. AppDelegate



Esta es una clase base propia de Cocos2d-x. Su función es la de gestionar el inicio y fin de la aplicación.

- *applicationDidFinishLaunching()*: se ejecuta cuando la aplicación termina de abrirse.
- *applicationDidEnterBackground()*: se ejecuta cuando la aplicación pasa a segundo plano.
- *applicationWillEnterForeground()*: se ejecuta cuando la aplicación vuelve activarse tras *applicationDidEnterBackground()*.

---

<sup>1</sup> Función típicamente llamada al arrancar un programa

### 3.1.2. MainMenu

MainMenu
+origin : Vec2 +visibleSize : Size - _background : Sprite* - _logo : Sprite* - _difficulty : short - _laps : short - _opponents : short
+createScene() : Scene* +init() : bool -createMenu() -playGame(pSender : Ref*) -showRanking(pSender : Ref*) -exitGame(pSender : Ref*)

Esta clase se corresponde con el menú principal del juego. Es por tanto la escena creada desde *AppDelegate*.

- *playGame(Ref\* pSender)*: crea una nueva instancia de *RaceConf* y reemplaza a ésta.
- *showRanking(Ref\* pSender)*: crea una nueva instancia de *RankingMenu* y reemplaza a ésta.
- *exitGame(Ref\* pSender)*: finaliza la aplicación.

### 3.1.3. RaceConf

RaceConf
+origin : Vec2 +visibleSize : Size - _background : Sprite* - _difficulty : short - _laps : short - _opponents : short
+createScene() : Scene* +init() : bool -createMenu() -startRace(pSender : Ref*) -backMainMenu(pSender : Ref*)

Esta clase corresponde a la escena de configuración de la carrera.

- *startRace(Ref\* pSender)*: recoge y guarda los parámetros introducidos, después crea una nueva instancia de *RaceScene* y reemplaza a ésta.
- *backMainMenu(Ref\* pSender)*: vuelve a *MainMenu*.

### 3.1.4. RankingMenu

RankingMenu
+origin : Vec2 +visibleSize : Size -_background : Sprite*
+createScene() : Scene* +init() : bool -createMenu() -backMainMenu(pSender : Ref*)

Esta clase corresponde a la escena de visualización del *ranking* de vueltas rápidas.

- *backMainMenu(Ref\* pSender)*: vuelve a *MainMenu*.

### 3.1.5. Race

Race
+origin : Vec2 +visibleSize : Size +audio : CocosDenshion::SimpleAudioEngine* +lapLabel : Label* +posLabel : Label* +playerPos : short +player : cocos2d::Sprite* +leftArrow : cocos2d::Sprite* +rightArrow : cocos2d::Sprite* -_tileMap : TMXTiledMap* -_tileAuxiliarMap : TMXTiledMap* -_speed : float -_difficulty : short -_timeStopped : short -_laps : short -_currentLap : short -_opponents : short -_currentPosition : short -_obstacles : Vector<Sprite*>
+createScene() : Scene* +init() : bool +update(delta : float) +createObstacle(delta : float) +carStopped(delta : float) +moveMap(delta : float) +checkLap(delta : float) -createControls(origin : Vec2, visibleSize : Size) -moveObstacles(v : Vector<Sprite*>) -deleteObstacle(s : Sprite*) -onTouchBegan(touch : Touch *, event : Event*) : bool -scheduleAll() -createMenu() -createLapLine() -updateLapsLabel() -updatePosLabel() -checkCollisions(v : Vector<Sprite*>) -checkDeletion(v : Vector<Sprite*>) -showEndRace(pSender : Ref*) -showRaceMenu(pSender : Ref*)

Esta clase es la escena principal del juego. Contiene todo lo visible en la escena de carrera.

- *\_difficulty*, *\_laps* y *\_opponents*: estas variables se recogen de la información introducida en *RaceConf*.
- *\_speed*: esta variable controla cómo de rápido se mueve el mapa y los rivales. Se calcula a partir de *\_difficulty* y *\_opponents*.
- *scheduleAll()*: este método programa (*schedule*) los métodos que deben ejecutarse cada *frame* o cada tiempo estimado.

- *update(float delta)*: bucle principal del juego. Se ejecuta cada *frame* y *delta* es el tiempo transcurrido entre el *frame* actual y el anterior (este tiempo es variable).  
Desde este método se realizan llamadas a *updatePosLabel()*, *moveObstacles(\_obstacles)*, *checkCollisions(\_obstacles)* y *checkDeletion(\_obstacles)*.
- *carStopped(float delta)*: éste método pasa a ser el bucle principal cuando el vehículo del jugador se para (ha colisionado con un obstáculo). Tras un tiempo llama de nuevo a *scheduleAll()* y se desactiva.
- *moveMap(float delta)*: mueve el mapa en función de *\_speed*.
- *moveObstacles(Vector<Sprite\*> v)*: mueve los obstáculos en función de *\_speed* para generar la sensación de que éstos están fijos en el mapa.
- *createObstacle(float delta)*: crea un obstáculo cada *delta* segundos en una posición semi-aleatoria.
- *checkLap(float delta)*: comprueba cada *delta* segundos si la vuelta actual es igual al número máximo de vueltas, para terminar la carrera.

### 3.1.6. RaceMenu

RaceMenu
+origin : Vec2 +visibleSize : Size - _background : Sprite*
+createScene() : Scene* +init() : bool -createMenu() -resumeRace(pSender : Ref*) -quitRace(pSender : Ref*)

Esta escena es el menú llamado desde *Race*.

- *resumeRace(Ref\* pSender)*: reanuda la carrera.
- *quitRace(Ref\* pSender)*: finaliza la carrera, crea una nueva instancia de *MainMenu* y se reemplaza por ésta.

### 3.1.7. EndRace

EndRace
-origin : Vec2 -visibleSize : Size - _background : Sprite*
+createScene() : Scene* +init() : bool -createMenu() -backMainMenu(pSender : Ref*)

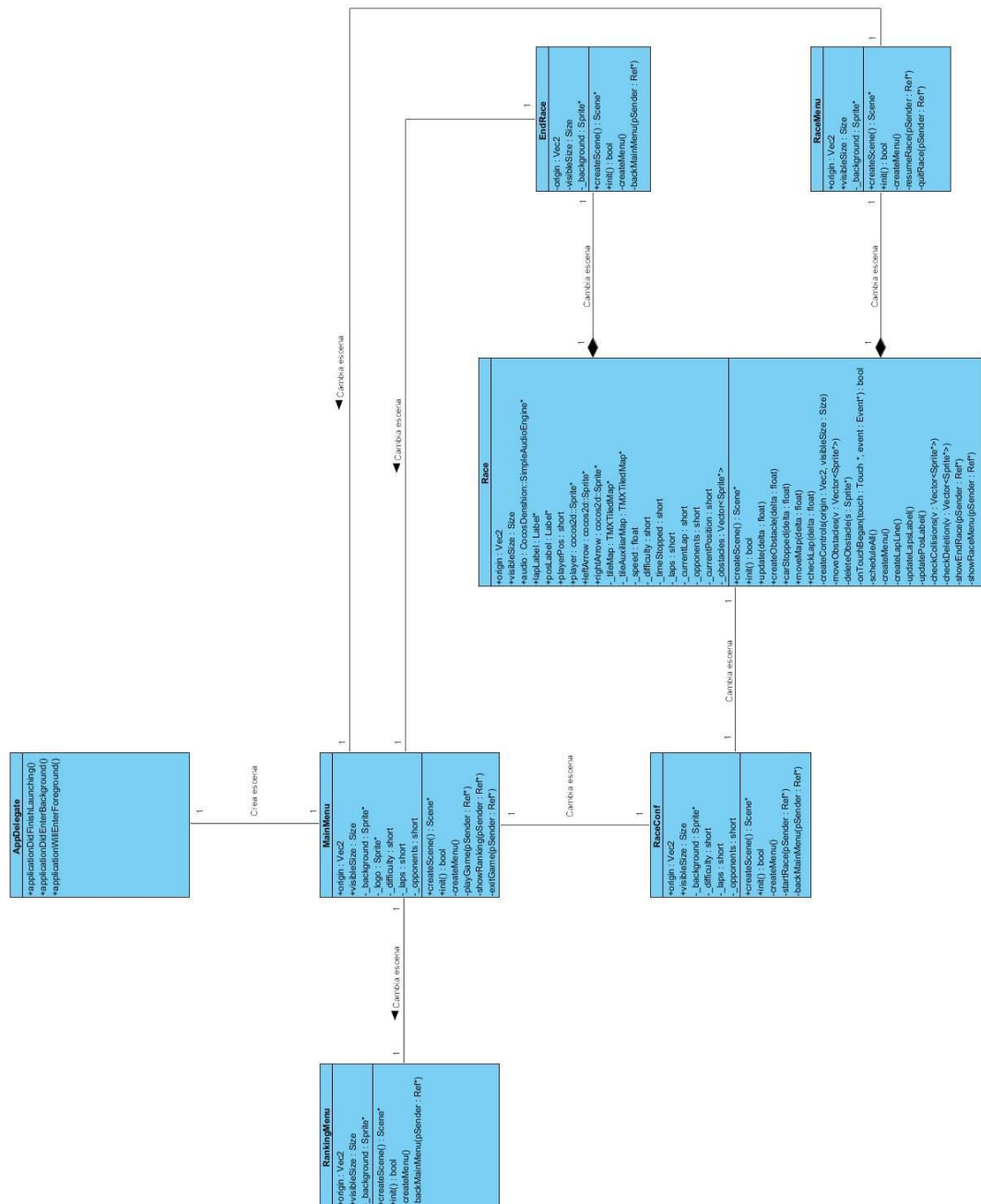
Esta es la escena que sigue a *Race* cuando la carrera termina. Muestra el tiempo obtenido y permite volver a *MainMenu*.

- *backMainMenu(Ref\* pSender)*: devuelve a *MainMenu*.

### 3.1.8. Diagrama de clases total

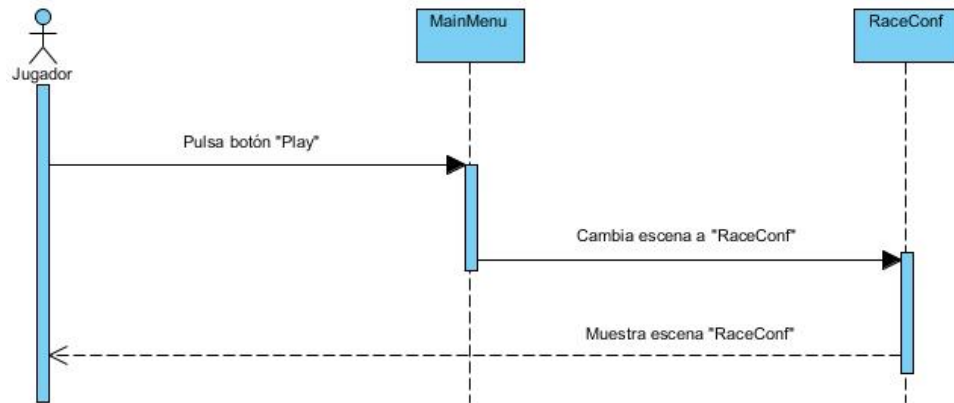
Este es el diagrama de clases total que finalmente refleja el diseño de la aplicación.

Es diferente del Modelo de Dominio (2.2) de la fase de *Análisis* pero mantiene la funcionalidad descrita.

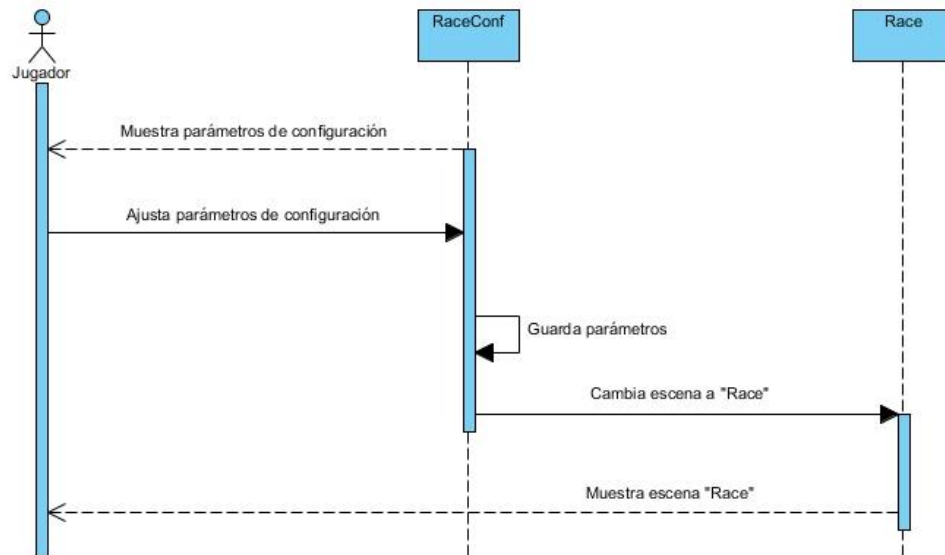


## 3.2. Diagramas de Secuencia del Sistema

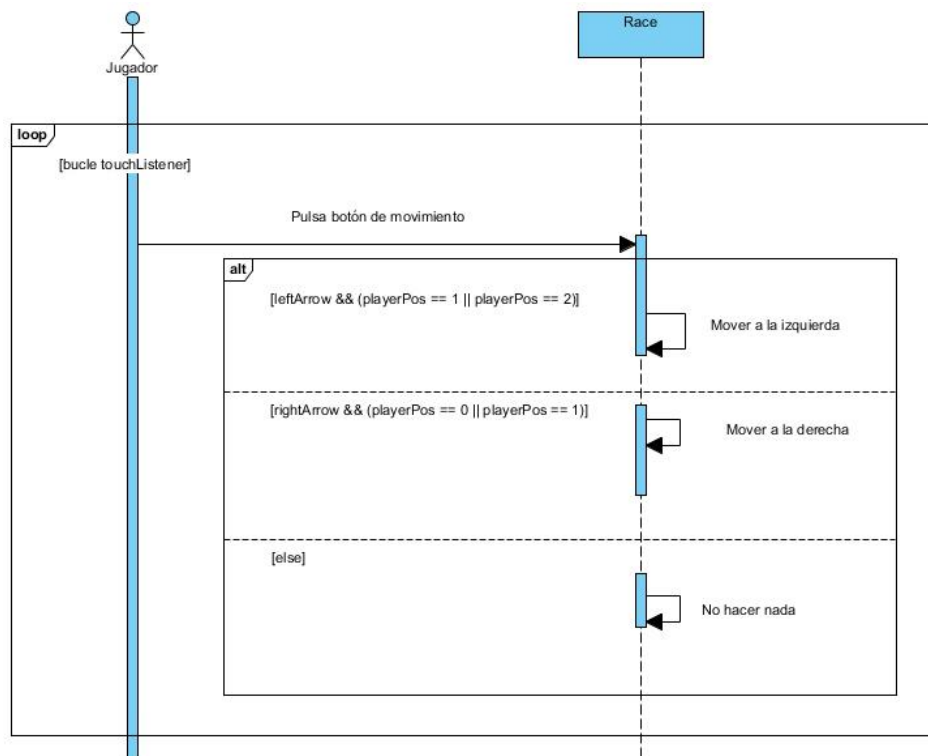
### 3.2.1. Iniciar partida



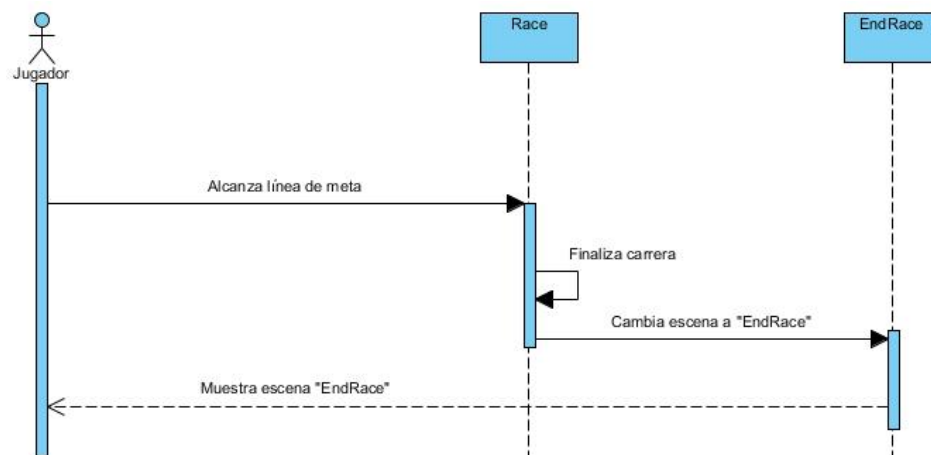
### 3.2.2. Iniciar carrera



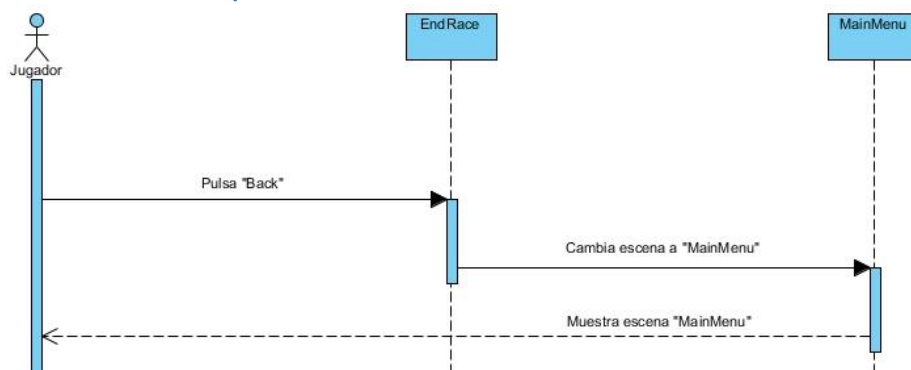
### 3.2.3. Mover vehículo



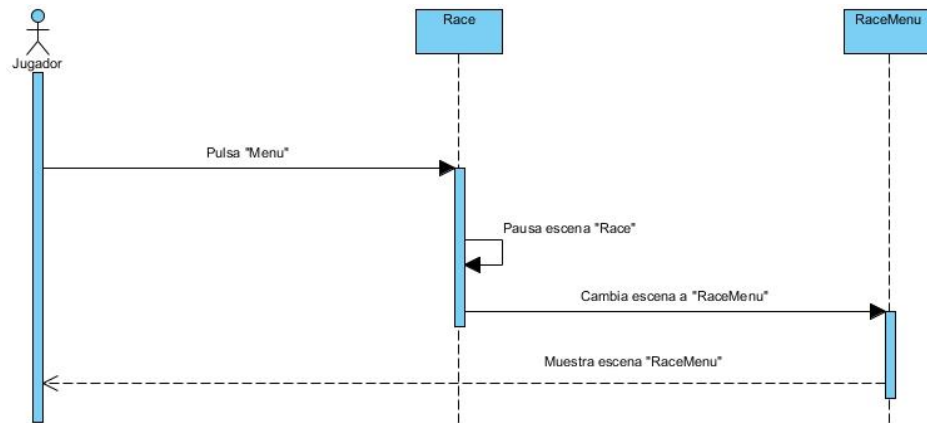
### 3.2.4. Terminar Carrera



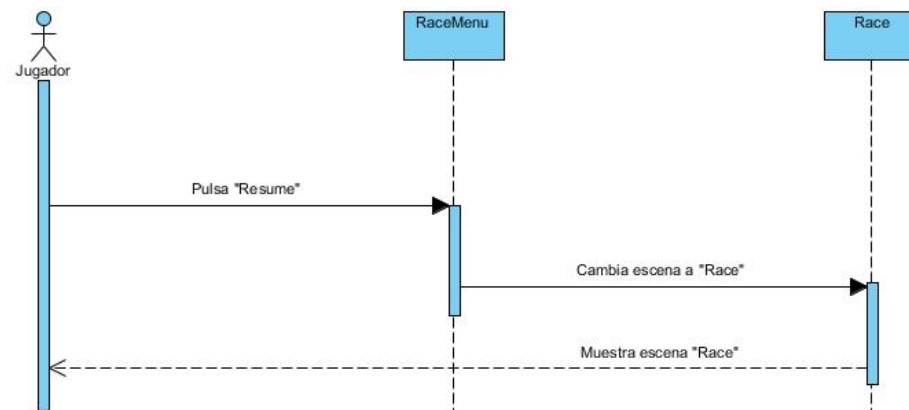
### 3.2.5. Finalizar partida



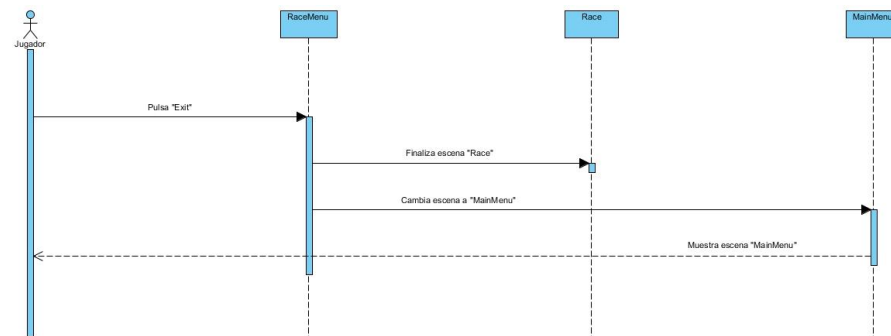
### 3.2.6. Pausar carrera



### 3.2.7. Reanudar carrera



### 3.2.8. Salir de carrera



### 3.2.9. Salir del juego

